(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 447 754 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
18.08.2004 Bulletin 2004/34

(51) Int Cl.?: **G06F 17/24**

(21) Application number: **04002224.6**

(22) Date of filing: **02.02.2004**

(84) Designated Contracting States:
AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HU IE IT LI LU MC NL PT RO SE SI SK TR
Designated Extension States:
AL LT LV MK

(30) Priority: **13.02.2003 US 366141**

(71) Applicant: **MICROSOFT CORPORATION**
Redmond, Washington 98052-6399 (US)

(72) Inventors:
- **Jones, Brian Michael**
  Redmond Washington 98052 (US)
- **Sawicki, Marcin**
  Kirkland WA 98033 (US)

(74) Representative: **Grünecker, Kinkeldey,**
Stockmair & Schwanhäusser Anwaltssozietät
Maximilianstrasse 58
80538 München (DE)

(54) Linking elements of a document to corresponding fields, queries and/or procedures in a database

(57) Methods and systems are provided for mapping and linking elements in a document to corresponding fields or queries in a database. A document is annotated with structural elements of a markup language, such as Extensible Markup Language (XML), in order to map portions of the document to the corresponding database. Once individual elements within the document are mapped and linked to corresponding data fields or queries within a selected database, changes made to individual elements within the document automatically cause updates to corresponding data in the database to which those elements are mapped and linked. Conversely, changes made to individual data fields within the selected database automatically update corresponding elements within the document.

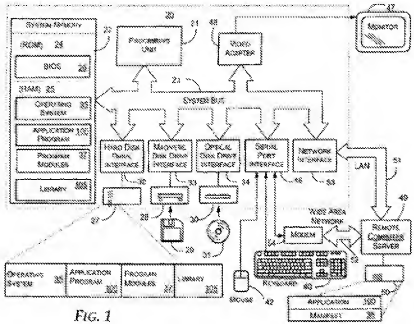

FIG. 1

EP 1 447 754 A1

Description

Field of the Invention

[0001] This invention relates generally to methods and systems for linking elements in a computer-generated document to corresponding data in a database.

Background of the Invention

[0002] Computer software applications allow users to create a variety of documents to assist them in work, education and leisure. For example, word processing applications allow users to create letters, articles, books, memoranda and the like. Spreadsheet applications allow users to store, manipulate, print and display a variety of alphanumeric data. Such applications have a number of well-known strengths including rich editing, formatting, printing, and calculation.

[0003] With the advent of modern databases, users are able to amass and manipulate large quantities of data associated with a variety of different subjects. Often, databases are located on a user's computer, or databases may be included on remote servers including remote Internet-based servers. Often many users may have access to a single database where each of the users add to, delete from, and manipulate data contained therein. For example, if a number of users constitute a project team developing a specification for a new type of computer software, each of the users may be assigned access to a shared document library contained on a given database. Accordingly, if a first user changes a section of the data contained in the specification, that change will be updated on the database and will be accessible by other users. Subsequently, another authorized user may gain access to the database to see changes made by the first user and to make additional changes or updates. Accordingly each of the users may develop and contribute to the data contained and managed in the database in a collaborative manner.

[0004] Often a user or group of users must assemble data from a given database into a letter, memorandum, article, spreadsheet, or other document for presenting the data to others. Continuing with the example described above, members of a software development team may be required at various points in a project to assemble the data contained in their shared database into a single document, such as a specification document, to present that document to future users of the new software or to a reviewer of the project. Unfortunately, once the document is prepared, the document becomes a static presentation of the data assembled from the database as it existed just before preparation of the document. If members of the project team update the data in the database after preparation of an initial draft of the document, the document must then be manually updated to reflect changes in the data upon which the document is based. Likewise, if during the preparation of the document, members make changes to data being placed in the document, the corresponding data in the database must then be manually updated to reflect changes made to the document that are not reflected in the data contained in the database.

[0005] Accordingly, there is a need for methods and systems for mapping and linking parts of document content to corresponding fields or queries in a database so that updates to the corresponding data in the database will automatically update corresponding parts of the document, and updates to parts of the document will automatically update the corresponding data in the database. It is with respect to these and other considerations that the present invention has been made.

Summary of the Invention

[0006] Embodiments of the present invention provide methods and systems for mapping and linking elements in a document to corresponding fields or queries in a database. A user of a computer-generated document such as a word processing document or a spreadsheet document associates various parts of the document with corresponding data in a database. According to one aspect of the invention, the document is annotated with structural elements of a markup language such as Extensible Markup Language (XML) in order to map portions of the document to the corresponding data in the database. If the document is marked up with XML structure, an XML scheme is attached to or associated with the document for setting the data types, data structures and XML elements rules for the document so that the user may annotate the document with the appropriate XML structure that adheres to the selected scheme. Alternatively, the user may utilize the document already containing XML structural annotation and already associated with an XML scheme.

[0007] After the document has been structured with one or more elements, a database is selected for association with the document. Selecting the database may include selecting a particular table within a document library maintained in the database where the particular table is associated with the document. After the particular table is selected within the database for association with the document, individual elements within the document are mapped to corresponding data fields or queries within the selected table. Queries may be represented as programs written in a database language such as SQL. Queries may be represented using stored procedures, in which case an element in the document may be mapped to a particular stored procedure in the database. Mapping the individual elements in the document to corresponding data fields or queries within the selected tables of the database may include providing a unique document identifier in the document to link (for data sending and receiving) each of the individual elements to a corresponding data field, query or procedure within the selected table or a stored procedure. Like-

wise, a unique identifier may be provided in the selected data fields, queries or stored procedures for linking individual data fields, queries or procedures to corresponding elements within the document.

[0008] Once individual elements within the document are mapped and linked to corresponding data within the selected table or database, changes made to individual elements within the document automatically cause updates to corresponding data fields, queries and/or procedures to which those elements are mapped and linked. If document elements are linked to queries providing the data for those elements, then they will typically also be linked to appropriate update queries that are able to update the database with the contents of the elements in the document in order to maintain a two-way link between the document and the database. Conversely, changes made to individual data fields, queries and/or procedures within the selected database tables automatically update corresponding elements within the document. If the elements are linked to queries, then when the elements in the document refresh their data, they will call the queries they are associated with for the latest set of results.

[0009] These and other features, advantages and aspects of the present invention may be clearly understood and appreciated from a review of the following detailed description of the disclosed embodiments and by reference to the appended drawings and claims.

Brief Description of the Drawings

[0010]

Fig. 1 is a block diagram of a computer and associated peripheral and networked devices that provide an exemplary operating environment for the present invention.

Fig. 2 is a simplified block diagram illustrating interaction between a document and a database where individual elements within the document are mapped and linked to corresponding data fields, queries and/or procedures within the database.

Fig. 3 illustrates a computer screen display of a software application for creating a document and for linking elements within the document to corresponding data fields, queries and/or procedures in a database.

Fig. 4 illustrates a computer screen display of a software application for creating a document and for linking elements within the document to corresponding data fields, queries and/or procedures in a database.

Fig. 5 is a simplified block diagram of a data field mapping user interface for allowing a user to map individual elements within a document to corresponding data fields, queries and/or procedures within a database.

Figures 6 and 7 are flow charts illustrating a method for mapping and linking elements of a document to corresponding data fields, queries and/or procedures of a database.

Detailed Description

[0011] The following description of embodiments of the present invention is made with reference to the above-described drawings wherein like numerals refer to like parts or components throughout the several figures. The present invention is directed to methods and systems for mapping and linking elements of a document to data fields, queries and/or procedures in a database.

Operating Environment

[0012] Fig. 1 and the following discussion are intended to provide a brief, general description of a suitable computing environment in which the invention may be implemented. While the invention will be described in the general context of an application program that runs on an operating system in conjunction with a personal computer, those skilled in the art will recognize that the invention also may be implemented in combination with other program modules. Generally, program modules include routines, programs, components, data structures, etc. that perform particular tasks or implement particular abstract data types. Moreover, those skilled in the art will appreciate that the invention may be practiced with other computer system configurations, including hand-held devices, multiprocessor systems, microprocessor-based or programmable consumer electronics, cell phones, minicomputers, mainframe computers, and the like. The invention may also be practiced in distributed computing environments where tasks are performed by remote processing devices that are linked through a communications network. In a distributed computing environment, program modules may be located in both local and remote memory storage devices.

[0013] With reference to Fig. 1, an exemplary system for implementing the invention includes a conventional personal computer 20, including a processing unit 21, a system memory 22, and a system bus 23 that couples the system memory to the processing unit 21. The system memory 22 includes read only memory (ROM) 24 and random access memory (RAM) 25. A basic input/output system 26 (BIOS), containing the basic routines that help to transfer information between elements within the personal computer 20, such as during start-up, is stored in ROM 24. The personal computer 20 further includes a hard disk drive 27, a magnetic disk drive 28, e.g., to read from or write to a removable disk 29, and an optical disk drive 30, e.g., for reading a CD-ROM disk 31 or to read from or write to other optical media. The hard disk drive 27, magnetic disk drive 28, and optical disk drive 30 are connected to the system bus 23 by a hard disk drive interface 32, a magnetic disk drive inter-

face 33, and an optical drive interface 34, respectively. The drives and their associated computer-readable media provide non-volatile storage for the personal computer 20. Although the description of computer-readable media above refers to a hard disk, a removable magnetic disk and a CD-ROM disk, it should be appreciated by those skilled in the art that other types of media which are readable by a computer, such as magnetic cassettes, flash memory cards, digital video disks, Bernoulli cartridges, and the like, may also be used in the exemplary operating environment.

[0014] A number of program modules may be stored in the drives and RAM 25, including an operating system 35, one or more application programs 100, a word processor program module 37 (or other type of program module), program data, such as the manifest 38, and other program modules (not shown).

[0015] A user may enter commands and information into the personal computer 20 through a keyboard 40 and pointing device, such as a mouse 42. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 21 through a serial port interface 46 that is coupled to the system bus, but may be connected by other interfaces, such as a game port or a universal serial bus (USB). A monitor 47 or other type of display device is also connected to the system bus 23 via an interface, such as a video adapter 48. In addition to the monitor, personal computers typically include other peripheral output devices (not shown), such as speakers or printers.

[0016] The personal computer 20 may operate in a networked environment using logical connections to one or more remote computers, such as a remote computer 49. The remote computer 49 may be a server, a router, a peer device or other common network node, and typically includes many or all of the elements described relative to the personal computer 20, although only a memory storage device 50 has been illustrated in Fig. 1. The logical connections depicted in Fig. 1 include a local area network (LAN) 51 and a wide area network (WAN) 52. Such networking environments are commonplace in offices, enterprise-wide computer networks, Intranets and the Internet.

[0017] When used in a LAN networking environment, the personal computer 20 is connected to the LAN 51 through a network interface 53. When used in a WAN networking environment, the personal computer 20 typically includes a modem 54 or other means for establishing communications over the WAN 52, such as the Internet. The modem 54, which may be internal or external, is connected to the system bus 23 via the serial port interface 46. In a networked environment, program modules depicted relative to the personal computer 20, or portions thereof, may be stored in the remote memory storage device. It will be appreciated that the network connections shown are exemplary and other means of establishing a communications link between the computers may be used.

Operation

[0018] Fig. 2 is a simplified block diagram illustrating interaction between a document and a database where individual elements within the document are mapped and linked to corresponding data fields, queries and/or procedures within the database. According to embodiments of the present invention, a user associates elements within a document with corresponding data fields, queries and/or procedures in a database. As shown in Fig. 2, an illustrative document 210 is shown for linking elements within the document to corresponding data fields, queries and/or procedures contained in a database 205. As should be understood by those skilled in the art, the document 210 is illustrative of documents that may be created and utilized from a variety of software applications including word processor applications, spreadsheet applications, web browser applications, and the like. For purposes of example only, the document 210 is illustrative of a word processing document wherein the user is preparing a resume having a title 212 and an education section 213. As should be understood by those skilled in the art, the text illustrated in Fig. 2 is for purposes of example only, and the data entered into the document may be in any type of format including alphanumeric text and images allowable by the software application under which the document is created.

[0019] In order to prescribe, map and link elements within the document 210 to corresponding data fields, queries and/or procedures in the database 205, elements within the document 210 are structurally annotated so that those elements may be identified and pointed to corresponding data fields, queries and/or procedures within the database 205. According to an embodiment of the present invention, structural annotation is provided to the document 210 using a markup language such as the Extensible Markup Language (XML). As shown in document 210, an XML element <title> element is applied to the document to provide structure for the title section 212 of the document and an <education> element 215 is applied to the document for providing structure to the education section 213 of the document.

[0020] Document elements are linked and mapped to corresponding data fields, queries and procedures in the database by providing a unique identifier (ID) number to the document and by associating the unique document ID with the a given record in a given table in the database. Accordingly, the correct record may be found in the database when a corresponding element in the document is modified and the correct element in the document may be found when a corresponding record in the database is modified. Preferably the unique ID number is stored in the document.

[0021] In order to provide the document with a set of

grammatical rules governing the type and structure of data that may be included in the document. As illustrated in Fig. 2, an XML schema is attached to or associated with the document for providing the rules governing each of the XML elements with which the user may annotate the document. For example, the resume document may have an attached or associated schema file 230 such as "resume-schema.xsd" for providing the allowable set of XML elements such as <title>, <education>, <experience>, <personal interest>, and so on. The schema file includes the rules governing the order with which those elements may be applied to the document and specific rules associated with individual elements applied to the document. For example, a schema attached to or associated with the resume document, illustrated in Fig. 2 may prescribe that data associated with the <education> element must include the name of a college or university followed by the address of the college or university.

[0022] As is understood by those skilled in the art, developers of XML schema files determine the names of XML elements and the associated data types and data structures allowed for those elements. Then, all users of documents annotated with XML structure according to a given schema file 230 may utilize the data contained within the XML structure without regard for the overall type and structure of the document. For example, if the resume document, illustrated in Fig. 2, is transmitted to a prospective employer, the prospective employer may develop software applications for parsing the document to locate specific types of data within the document for use by the prospective employer. The prospective employer, for example, may wish to create a database of colleges and universities from which prospective employees have graduated. Using the schema file 230 attached to the document, the prospective employer will know that the data associated with the <education> element has been prepared according to the schema file governing the document. Accordingly, the prospective employer may develop a software application for locating the <education> element and for extracting the data associated therewith for insertion into the prospective employer's database. As set forth above, in order to link the <education> element, for example, with an education record in the database, a unique identifier (ID) number must be stored in the document for associating the document and its elements with corresponding records in the database.

[0023] The prospective employer, according to this example, may extract this data without regard to other aspects of the document such as the location of and data contained within the title section. This is made possible by the fact that each user of the document follows the data type and data structure rules prescribed in the schema file attached to or associated with the document. The schema file 230 may be attached to the document, or the schema file may be maintained in a separate location such as a library of schema files accessible by the document. That is, the document may contain a file path pointer or a unique name space identifier (e.g., URI or URN) for locating and/or identifying the schema file 230 for providing the document rules governing the XML structure of the document.

[0024] As briefly described above, elements of the document, such as the <title> and <education> elements may be linked to data fields, queries and/or procedures within a data base 205 so that information updated in corresponding data fields, queries and/or procedures will automatically update data contained in corresponding elements contained within the document, and vice versa as described below.

[0025] Updates to the database may be done through various queries, for example using the structured query language (SQL), that affect multiple fields and tables in the database. In addition to coming directly from individual data fields, data in the document may come from the results of a query that assembles information from many fields and/or tables of the database based on various selection criteria. Additionally, as is understood by those skilled in the art, many databases support the use of stored procedures that are programs for executing commands against the database for writing data to the database, extracting data from the database and for manipulating data in the database.

[0026] In addition to associating data fields with elements in a document, queries and stored procedures may be associated with elements in the document. Two types of queries may be created for each element in order for the link between the element and the query to be "two-way" (read-write), as opposed to "one-way" (read-only). If only one query is provided for reading data from the database, then changes to the data can only be made in the database allowing the document to only read them. On the other hand, if only one query is provided that updates the database based on the data in the document, then only the document can be changed to update the database. A two-way link is created if two queries are provided where one query is for reading data from the database and one query is for writing into the database. Accordingly, the document or the database may be updated and the other will inherit those updates automatically. In some embodiments, it may be preferable to have only one-way linking. For example, a database administrator may prefer that changes to a particular data field may only be made by updating the database and not the document. Then only a query that reads the latest data from the database would be necessary to keep the document up-to-date with respect to the database.

[0027] For example, suppose the user is a patent attorney whose task is to author patent applications. Each patent application at his/her firm is assigned a unique document identifier (ID) number. Also suppose his firm uses a database to store the associations between patent attorney names and the patent applications assigned to each attorney. If he/she has been assigned a

particular patent application document, then the database contains a record identifying him/her as the owner of that patent application document by storing his/her name and the document ID in the same data record in a given database table. In addition to this information being stored in the database, the format of the application document itself requires the document ID and the name of the attorney to appear in it. Without this invention, the name of the attorney would have to be entered into the database, and if the attorney name changes in the database (for example, the application is re-assigned to another attorney in the firm), the application document would have to be manually edited to replace the previous attorney name with the new attorney name.

[0028] According to an embodiment of the present invention, a means is provided for inserting into the document an appropriate association with the corresponding data in the database, so that if the corresponding data record in a given database table changes (for example, a different attorney gets assigned to the application document), the document will automatically reflect this change and update the attorney name appearing in the document. If the association is "two-way", meaning that a way for the document to update the database is also specified, then the attorney name may be changed in the document and the change will be updated in the database. The "two-way" association or communication is created is by marking up the attorney name in the document with an appropriate markup element (for example, <AttorneyName> if XML is used) and specifying a mapping between that element and the appropriate query in the database (for example, represented by a stored procedure in the database called "GetCurrentAttorneyName" for reading from the database, and another procedure called "SetCurrentAttorneyName" for writing into the database.) This association itself may be stored in the document or in some part of a program module associated with the document.

[0029] At the database 205, data fields, queries and/or procedures corresponding to prescribed elements contained within the document 210 may be created for receiving, storing, sorting, and maintaining data associated with those elements. For example, a user of the resume document 210, illustrated in Fig. 2, may prepare a database record in the database 205 containing a data field associated with the <education> element 215 for maintaining data to be placed in the education section 213 of the resume document 210. As should be understood, the database may be organized according to a variety of logical associations of data. A particular data, query or procedure may be stored in a particular field. A collection of fields associated with a document may be assembled in a database record. A database table may include a number of records associated with a class of documents.

[0030] Once the corresponding data field is established in the database 205, the unique document ID, described above is written to the document 210 and associated with the <education> element to point that element to the corresponding record in the database 205 containing the data field with the data for the <education> element. Likewise, unique document ID is used to link the data field containing education data back to the <education> element of the document 210. Accordingly, when data in the education field of the database (205) record corresponding to this particular file is updated, the data contained in the education section 213 of the resume document 210 is automatically updated. Conversely, if the user changes the data contained in the education section 213 of the document 210, those changes are automatically sent to the education data field of the appropriate record in the database 205 to update data contained therein.

[0031] As should be understood by those skilled in the art, a software application program module may be written to both the document software application and the database software application for calling the corresponding data fields, queries and/or procedures or for calling the corresponding document element to request updates to the corresponding data field or to the corresponding document elements data when data is changed in either the database or the document. The software program module for directing the communication between the document element and the corresponding data field and vice versa may be a software program module written to the document application and/or the database application, or the program module may operate as an application programming interface or dynamic-link library accessible by the document application and/or database application. The database application and corresponding memory for the database 205 may be located remotely from the user's computer 20 on a remote computer server 49 accessible to the user's computer via internet-based web server or database server or via an Internet connection to a remote database server.

[0032] According to an embodiment of the present invention, the database 205 may contain a document library 220 in which a variety of prescribed document types may be maintained along with associated data fields, queries and/or procedures. As will be described below, a user may select a document type from the document library 220 via the user's document application for opening a particular document that is already structurally annotated and associated with corresponding data fields, queries and/or procedures within the database. For example, if the user is a member of a project team preparing a patent application specification document, the patent application specification document may be contained in the document library 220. When the user desires to work on the patent application specification document, the user may select the patent application specification document from the document library via the database 205. The patent application specification document opens to the user using the user's document application with all structural annotation to the document

already in place. For example, the patent application specification document may be in the form of a template containing XML markup annotation and an associated XML schema and file path pointers for allowing the user to begin preparation of the document from data downloaded from corresponding data fields, queries and/or procedures in the database 205, or the user may prepare the document by inserting data into the document that will then automatically be sent to corresponding data fields, queries and/or procedures contained within the database 205.

[0033] As shown in Fig. 2, upon selection of a given document from the document library 220 on the database side, the user may access the data fields queries and/or procedures for the resume document associated with the resume document 210. Accordingly, the user may update data contained in the individual data fields, queries and/or procedures 225 in order to cause the data automatically to be updated in the corresponding document 210. The database 205 may be operated in a shared data environment where a number of users may have access to a single database such as the resume document database 225 for adding to, deleting from, and generally updating data contained in the corresponding data fields, queries and/or procedures. Because each data field such as the title data field in the resume document 225 is mapped to and linked to the corresponding title section 212 of the resume document 210, changes to the data contained in the title data field of the resume document 225 will cause an update of the information contained in the title section 212 of the document. For example, if the data in the title section of the resume document 225 is changed from "John Doe" to "Jane Doe", the information contained in the title section of the resume document 210 will automatically be changed from "John Doe" to "Jane Doe". Likewise, if the user opens the resume document 210 and changes the title from "John Doe" to "Jane Doe", the data contained in the title section of the resume document data field 225 at the database 205 likewise will be updated automatically.

[0034] Fig. 3 illustrates a computer screen display of a software application for creating a document and for linking elements within the document to corresponding data fields, queries and/or procedures in a database. According to one embodiment of the present invention, and as described briefly above, a document library 220 may be accessed at the database 205 for obtaining a previously created document that is mapped to a corresponding database or for obtaining a document template for creating a document that may be mapped to a corresponding database. Referring to Fig. 3, upon selection of an exemplary document library button 310 of the user's document application 300, a document library user interface 320 may be launched for providing the user with a list of available documents or document types. For example, the user may select the resume document 325 from the document library user interface 320 to launch the resume document 210 illustrated in Fig. 2.

[0035] Once the user launches the resume document 210 by selection of the resume document from the document library user interface 320, the resume document 210 is displayed to the user for editing. According to one embodiment of the present invention, the resume document launched may include the most recent version of the resume document 210 including data populated in each of the document elements from the corresponding data fields, queries and/or procedures of the database 205 because document elements are associated with corresponding database records by matching the unique document ID with corresponding database records. As described above with reference to Fig. 2, once the user updates data contained in the various elements of the resume document 210, the data contained in the corresponding data fields, queries and/or procedures of the database 205 is updated. Likewise, changes to data contained in corresponding data fields, queries and/or procedures of the database 205 will automatically update corresponding data contained in data elements of the resume document 210.

[0036] Alternatively, documents listed in the document library using interface 220 may include a variety of template documents accessible by the user for associating with corresponding data fields, queries and/or procedures in the database 205. According to an embodiment of the invention, each document type may include a number of structural elements such as XML elements to form a template for the desired document. After the user has completed the selected document, for example completing the education section of the resume document 210, the user may select a data location on the database 205 containing data fields, queries and/or procedures corresponding to the preformatted data elements of the selected document. Accordingly, when the user saves the prepared document, data inserted into the document will also be saved into corresponding data fields, queries and/or procedures in the database 205. Then, as described above, any time the corresponding data fields, queries and/or procedures in the database are changed or updated, the corresponding elements in the document will likewise be changed or updated and vice versa. If a new document is created and saved to the document library or database, a new document ID may be generated for it automatically so that a record corresponding to that document can be created in the database.

[0037] Referring to Fig. 4, and according to an embodiment of the present invention, a document that is not presently mapped or linked to a corresponding database may be structured in order to link various elements of the document to corresponding data fields, queries and/or procedures in the database 205. For example, if the user prepares a document such as a resume document 210, by preparing the document in the data entry area 305 of his/her word processor 300,

the user may select a resume document type from the document library user interface 320, described with reference to Fig. 3, in order to provide the user with a suggested list of elements to apply to the document being prepared by the user. As shown in Fig. 4, in response to selection of the resume document type, a suggested resume element pane 350 may be provided to the user to provide the user suggested elements for annotating the resume document with XML structure. According to one embodiment of the present invention, the user may enter the document and manually insert XML elements such as the <education> element 360, or the user may place his cursor 365 within the education section of the document and select the education element 365 from the pane 350 to automatically annotate the selected area of the document with the <education> element. As the user annotates the document with XML structure, an XML tree view pane 340 may be provided to show the user in outline form the XML structure applied to the document.

[0038] Once the newly created document is annotated with structure, such as XML element structure, the document elements may be linked to corresponding data fields, queries and/or procedures to allow communication between the document and the database 205, as described above. According to one embodiment of the present invention, the suggested elements provided to the user for annotating the document may be prepopulated with pointers to a corresponding data fields, queries and/or procedures in the database 205. Accordingly, annotation of the document with one of the suggested elements not only provides the desired structure to the document, but points the associated element to the corresponding data field in the database 205.

[0039] Alternatively, referring to Fig. 5, a data field mapping user interface 500 may be provided to the user for mapping elements of the document with corresponding data fields, queries and/or procedures in the database 205. As shown in Fig. 5, a list of elements from the document may be populated into the user interface 500 along with an associated list of data fields, queries and procedures to which the document elements may be mapped and linked. As should be understood the user interface 500 is equally applicable for mapping document elements to queries 540 and procedures 545. That is, as is illustrated in Fig. 5, the user interface may be extended to include queries (for example, SQL statements) or names of stored procedures for data reading and writing between the document and the database.

[0040] For example, if the user desires that data inserted into the title section of the document should be mapped and linked to the title data field in the database 205, the user may select the <title> element followed by selection of the title data field in order to map and link the title element of the document to the title data field of the database 205. Accordingly, after mapping the title element to the title data field future changes to data contained in either the title section of the document or the title data field of the database 205 will cause changes in the corresponding document element or data field and vice versa. Once all desired document elements are mapped and linked to corresponding data fields, queries and/or procedures, and a unique ID for associating the document with a record or an appropriate set of data exists, data communication between document elements and corresponding data fields, queries and/or procedures is established.

[0041] Figures 6 and 7 are flow charts illustrating a method for mapping and linking elements of a document to corresponding data fields, queries and/or procedures of a database. The method of 600 begins at start step 605 and moves to step 610 where a database or table 205 is established for maintaining and manipulating data. For purposes of discussion of Figures 6 and 7, assume for example that a table is established at the database 205 to maintain data used in preparation for an eventual patent application specification document. At step 615, a determination is made as to whether the user must create a new document. If not, the method proceeds to method 635 and a schema such as "patent-specification/document-schema.xsd" may be attached to an existing patent application specification document to provide the rules and procedures available for annotating the document with XML structure. At step 640, the schema is attached to the existing document. Alternatively, the document may already have an attached or associated schema.

[0042] If at step 615 a determination is made that a new document must be created, the method proceeds to step 620, and the user creates a new patent specification document and stores in the document a unique document ID for linking document elements to records in the database, as described above with reference to Figures 3, 4 and 5. At step 625, a check of the database 205 is performed to determine whether a schema for the new document being created by user is available. As discussed above with reference to Figures 3 and 4, this determination may be made by selecting the document library to determine whether the document library at the database 305 includes a document type that may be associated with the new document being created by the user. For example, as shown in Fig. 3, the user may select the patent disclosure document type from the document library user interface 320, and at step 630 the schema associated with the patent disclosure document type may be obtained and attached to the new document being prepared by the user.

[0043] At step 645, the document being created and/or selected by the user is annotated with XML elements as desired by the user. As should be understood, if the user has selected an existing document at step 615, no additional structural annotation may be required to the document. At step 650, the user may specify a table within the database 205 for associating the document elements with corresponding data fields, queries and/or procedures maintained in that table of the database 205.

At step 655, as described above with reference to Figures 4 and 5, elements in the document are mapped and linked to corresponding data fields, queries and/or procedures within a given table in the database 205 in order to facilitate data communications between elements in the document and corresponding data fields, queries and/or procedures in the database 205.

[0044] Referring back to step 645, if a table has not been prepared at the database 205 for maintaining data associated with the document being created by the user, the user may specify XML elements included in the document to have corresponding data fields, queries and/or procedures within a selected table. For example, the database 205 may contain many tables in which a number of data fields, queries and/or procedures or document types may be included. The table may be established within the database 205 for maintaining data associated with patent specification documents. Within the table created for patent specification documents, a variety of subtables may be created for maintaining data for individual patent specification documents. Within each subtable, a variety of data fields, queries and/or procedures may be created for associating with individual elements contained within the patent specification document being created by the user.

[0045] At step 660, individual data fields, queries and/or procedures within the selected table may be established for associating with XML elements applied to the document. According to one embodiment, annotation of the document and mapping of the document to suggested data fields, queries and/or procedures, as described above with reference to Fig. 6, establishes the corresponding data fields, queries and/or procedures within the selected table. Alternatively, at step 665, the user may enter the database 205 directly and create a table with data fields, queries and/or procedures that may be mapped to selected document elements as applied to the document being created by the user.

[0046] Once the document is prepared and annotated with XML structure, and once the data fields, queries and/or procedures for containing data corresponding to the document elements are established, the method proceeds to step 670. Figure 7, where regular usage of the table and document may begin. At step 675, if the user changes and saves the document, the method proceeds to step 680 and data changed and saved in various elements of the document is updated at the table by updating corresponding data fields, queries and/or procedures in the table. On the other hand, at step 685, if the user makes changes directly to the data contained in the data fields, queries and/or procedures corresponding to elements in the document, the method proceeds to step 690 and data associated with corresponding data elements in the document is automatically updated as the corresponding data is changed in the corresponding data fields, queries and/or procedures. The method ends at step 695.

[0047] As described above, methods and systems are provided for mapping and linking elements of the document to corresponding data fields, queries and/or procedures in a database. It will be apparent to those skilled in the art that various modifications or variations may be made in the present invention without departing from the scope or spirit of the invention. Other embodiments of the invention will be apparent to those skilled in the art from consideration of the specification and practice of the invention disclosed herein.

Claims

1. A method of linking elements in a computer-generated document to corresponding data in a database, comprising:

   applying elements of a markup language to the document;
   linking one or more markup language elements in the document to corresponding data in the database;
   entering data into the database associated with a given markup language element in the document; and
   in response to entering data into the database associated with the given markup language element in the document, automatically writing the data to the document in a location in the document associated with the given markup language element.

2. The method of claim 1, further comprising establishing data fields within the database for linking to corresponding markup language elements in the document.

3. The method of claim 2, further comprising writing a unique document identifier to the document for linking the data fields in the database to the document.

4. The method of claim 3, further comprising:

   writing a database query to the database for assembling data from one or more data fields within the database; and
   writing the results of the database query into the document in a location in the document associated with the database query.

5. The method of claim 4, further comprising associating the database query with a given markup language element in the document for writing the results of the database query into the document in a location in the document associated with the database query.

6. The method of claim 5, further comprising storing a

programming procedure in the database for reading the data from the database and for writing the data to the document in a location in the document associated with the given markup language element.

7. The method of claim 6, further comprising
updating the results of the database query when data in the database associated with the database query is updated; and
executing the programming procedure when results of the database query are updated.

8. The method of claim 7, whereby the procedure is in the format "GetCurrentMarkupElementData."

9. The method of claim 3, further comprising:
entering data into the document associated with a given markup language element; and
in response to entering data into the document associated with the given markup language element, automatically writing the data entered into the document to a data field in the database linked to the given markup language element.

10. The method of claim 9, further comprising
writing a database query to the database for writing data entered into the document to a data field in the database linked to the given markup language element.

11. The method of claim 10, further comprising associating the database query with a given markup language element in the document for writing data entered into the document to a data field in the database linked to the given markup language element.

12. The method of claim 11, further comprising storing a programming procedure in the database for writing the data entered into the document to a data field in the database linked to the given markup language element as required by the query

13. The method of claim 12, whereby the procedure is in the format "SetCurrentMarkupElementData."

14. The method of claim 1, prior to the step of applying elements of a markup language to the document, attaching a schema to the document defining rules associated with a markup language to be applied to the document

15. The method of claim 14, whereby the markup language is the Extendable Markup Language

16. The method of claim 14, whereby the markup language is the Hypertext Markup Language

17. The method of claim 1, whereby the step of linking one or more markup language elements in the document to corresponding data fields in the database further comprises:
providing a list of markup language elements contained in the document;
providing a list of data fields established for linking to corresponding markup language elements in the document;
selecting a markup language element from the list of markup language elements;
selecting a data field from the list of data fields for linking the selected data field to the selected markup language element, and
upon selection of the data field from the list of data fields for linking the selected data field to the selected markup language element, linking the selected data field to the selected markup language element.

18. A method of linking elements in a computer-generated document to corresponding data in a database, comprising:
applying elements of a markup language to the document;
linking one or more markup language elements in the document to corresponding data in the database;
writing a unique document identifier to the document for linking the one or more markup language elements in the document to corresponding data in the database;
entering data into the database associated with a given markup language element in the document;
in response to entering data into the database associated with the given markup language element in the document, automatically writing the data to the document in a location in the document associated with the given markup language element;
entering data into the document associated with a given markup language element; and
in response to entering data into the document associated with the given markup language element automatically writing the data entered into the document to a data field in the database linked to the given markup language element.

19. The method of claim 18, further comprising establishing data fields within the database for linking to corresponding markup language elements in the document.

20. The method of claim 19, further comprising:

writing a first database query to the database for assembling data from one or more data fields within the database and for writing the results of the first database query into the document in a location in the document associated with the database query;
writing a second database query to the database for writing data entered into the document to a data field in the database linked to the given markup language element.

21. The method of claim 20, further comprising associating the first and second database queries with a given markup language element in the document.

22. The method of claim 21, further comprising storing a first programming procedure in the database for reading the data from the database and for writing the data to the document in a location in the document associated with the given markup language element.

23. The method of claim 22, further comprising
updating the results of the database query when data in the database associated with the database query is updated; and
executing the first programming procedure when results of the database query are updated.

24. The method of claim 23, further comprising storing a second programming procedure in the database for writing the data entered into the document to a data field in the database linked to the given markup language element as required by the query.

25. The method of claim 24, whereby the markup language is the Extendable Markup Language.

26. The method of claim 25, whereby the markup language is the Hypertext Markup Language.

27. The method of claim 18, whereby the step of linking one or more markup language elements in the document to corresponding data fields in the database further comprises:

providing a list of markup language elements contained in the document;
providing a list of data fields established for linking to corresponding markup language elements in the document;
selecting a markup language element from the list of markup language elements;
selecting a data field from the list of data fields for linking the selected data field to the selected markup language element; and
upon selection of the data field from the list of data fields for linking the selected data field to the selected markup language element, linking the selected data field to the selected markup language element.

28. A computer readable medium having stored thereon computer-executable instructions which when executed by a computer, perform the steps of claim 18.

29. A method of linking elements in a computer-generated document to corresponding data fields in a database, comprising:

providing a communication link between one or more markup language elements in the document to corresponding data fields in the database;
entering data into the document associated with a given markup language element; and
in response to entering data into the document associated with the given markup language element, automatically saving the data to a data field in the database corresponding to the given markup language element.

30. The method of claim 29, prior to the step of providing a communication link between one or more markup language elements in the document to corresponding data fields in the database, selecting the document from a document library containing documents annotated with the one or more markup language elements and the document associated with the database, the data base containing data fields established for linking to the one or more markup language elements.

31. The method of claim 30, whereby the step of providing a communication link between one or more markup language elements in the document to corresponding data fields in the database further comprises:

providing a list of markup language elements contained in the document;
providing a list of data fields established for linking to corresponding markup language elements in the document;
selecting a markup language element from the list of markup language elements;
selecting a data field from the list of data fields for linking the selected data field to the selected markup language element; and
upon selection of the data field from the list of data fields for linking the selected data field to the selected markup language element, linking the selected data field to the selected markup language element.

32. A method of claim 31, whereby the step of linking the selected data field to the selected markup language element includes writing a unique document identifier to the document and associating the unique document identifier with the selected markup language element and associating the unique document identifier with the selected data field to point the selected markup language element to the selected data field in the database.

33. A computer readable medium having stored thereon computer-executable instructions which when executed by a computer, perform the steps of:

linking one or more XML elements in a document to corresponding data fields in a database;
entering data into the document associated with a given XML element; and
in response to entering data into the document associated with the given XML element, automatically saving the data to a data field in the database corresponding to the given XML element.

34. The computer readable medium of claim 33, prior to the step of linking one or more XML elements in the document to corresponding data fields in the database, selecting the document from a document library containing documents annotated with the one or more XML elements and the document associated with the database, the data base containing data fields established for linking to the one or more XML elements.

35. The computer readable medium of claim 34, whereby the step of linking one or more XML elements in the document to corresponding data fields in the database further comprises:

providing a list of XML elements contained in the document;
providing a list of data fields established for linking to corresponding XML elements in the document;
selecting an XML element from the list of XML elements;
selecting a data field from the list of data fields for linking the selected data field to the selected XML element; and
upon selection of the data field from the list of data fields for linking the selected data field to the selected XML element, linking the selected data field to the selected XML element.

36. The computer readable medium of claim 35, whereby the step of linking one or more XML elements in the document to corresponding data fields in the database includes writing a unique document identifier to the document and associating the unique document identifier with the selected XML element and associating the unique document identifier with the selected data field to point the selected XML element to the selected data field in the database.

37. The computer readable medium of claim 36 having stored thereon computer-executable instructions which when executed by a computer, further perform the step of:

entering data into a given data field;
saving the data to the given data field; and
in response to saving the data to the given data field, automatically saving the data to the document in a location associated with a given XML element that corresponds to the given data field.

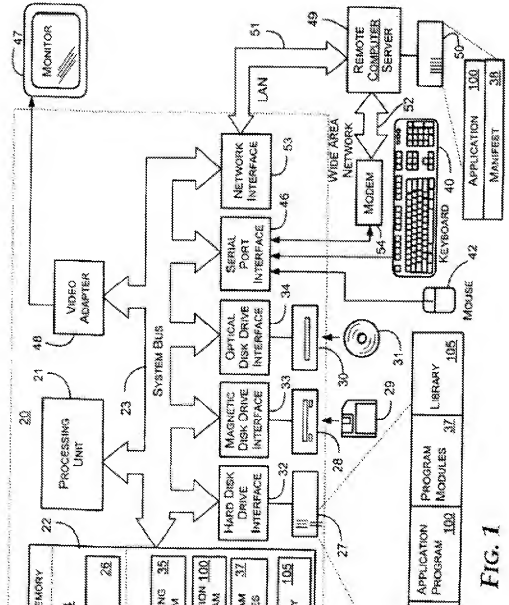
20
System Memory
22
(ROM) 24
BIOS 26
(RAM) 25
Operating System 35
Application 100 Program
Program 37 Modules
Library 105
Processing Unit
21
System Bus
23
Video Adapter
48
Monitor
47
Hard Disk Drive Interface
32
Magnetic Disk Drive Interface
33
Optical Disk Drive Interface
34
Serial Port Interface
46
Network Interface
53
LAN
51
Wide Area Network
Modem
54
Remote Computer Server
49
Application 100
Manifest 38
Keyboard
40
Mouse
42
27
28
29
30
31
52
50
Operating System 35
Application Program 100
Program Modules 37
Library 105

FIG. 1

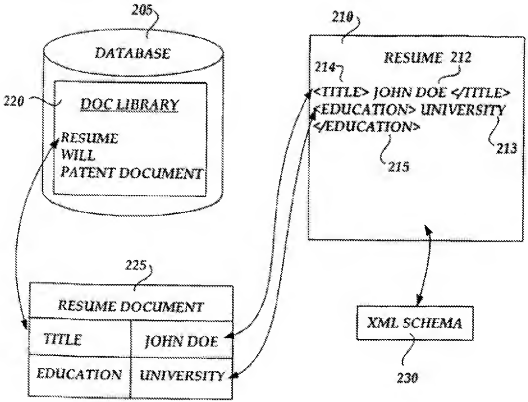
205
DATABASE
220
DOC LIBRARY
RESUME
WILL
PATENT DOCUMENT
210
RESUME 212
214
<TITLE> JOHN DOE </TITLE>
<EDUCATION> UNIVERSITY
</EDUCATION>
213
215
225
RESUME DOCUMENT
TITLE
JOHN DOE
EDUCATION
UNIVERSITY
XML SCHEMA
230

Fig. 2

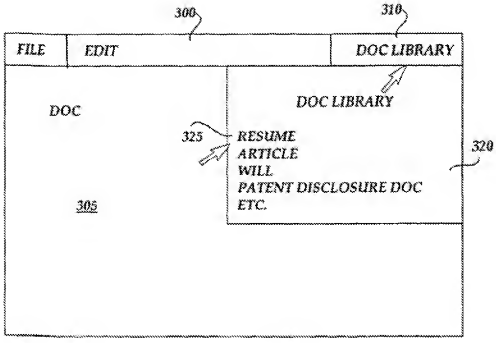
300
310
FILE
EDIT
DOC LIBRARY
DOC LIBRARY
DOC
325
RESUME
ARTICLE
WILL
PATENT DISCLOSURE DOC
ETC.
320
305

Fig. 3

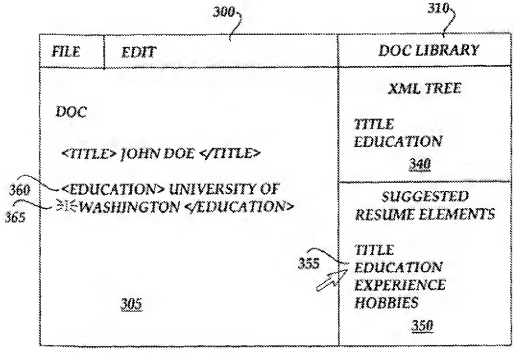
300
310
FILE
EDIT
DOC LIBRARY
XML TREE
DOC
TITLE
EDUCATION
340
<TITLE> JOHN DOE </TITLE>
360
<EDUCATION> UNIVERSITY OF
365
WASHINGTON </EDUCATION>
SUGGESTED
RESUME ELEMENTS
355
TITLE
EDUCATION
EXPERIENCE
HOBBIES
305
350

Fig. 4

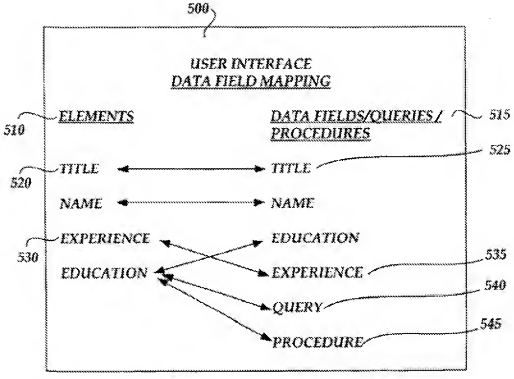
500
USER INTERFACE
DATA FIELD MAPPING
ELEMENTS
510
DATA FIELDS/QUERIES / PROCEDURES
515
TITLE
520
TITLE
525
NAME
NAME
EXPERIENCE
530
EDUCATION
EDUCATION
EXPERIENCE
535
QUERY
540
PROCEDURE
545

Fig. 5

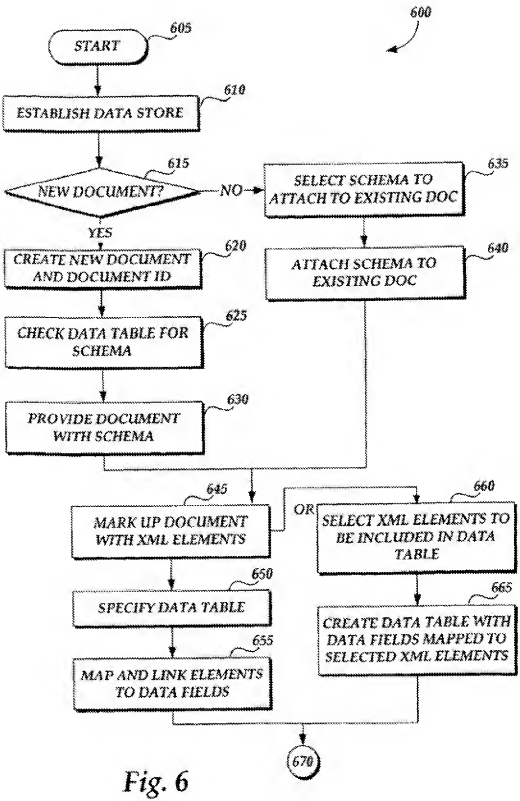
600
START
605
ESTABLISH DATA STORE
610
NEW DOCUMENT?
615
NO
SELECT SCHEMA TO ATTACH TO EXISTING DOC
635
YES
CREATE NEW DOCUMENT AND DOCUMENT ID
620
ATTACH SCHEMA TO EXISTING DOC
640
CHECK DATA TABLE FOR SCHEMA
625
PROVIDE DOCUMENT WITH SCHEMA
630
MARK UP DOCUMENT WITH XML ELEMENTS
645
OR
SELECT XML ELEMENTS TO BE INCLUDED IN DATA TABLE
660
SPECIFY DATA TABLE
650
CREATE DATA TABLE WITH DATA FIELDS MAPPED TO SELECTED XML ELEMENTS
665
MAP AND LINK ELEMENTS TO DATA FIELDS
655
670

Fig. 6

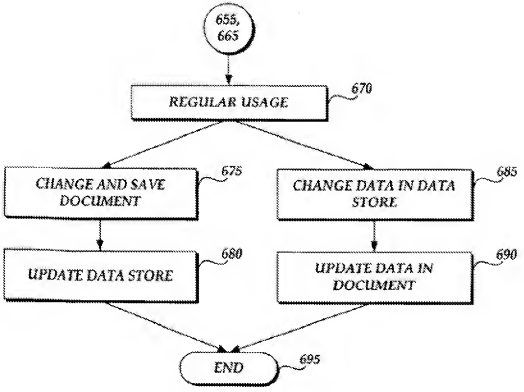
655, 665
REGULAR USAGE
670
CHANGE AND SAVE DOCUMENT
675
CHANGE DATA IN DATA STORE
685
UPDATE DATA STORE
680
UPDATE DATA IN DOCUMENT
690
END
695

Fig. 7

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 04 00 2224

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
X	FERNANDEZ M ET AL: "SilkRoute: trading between relations and XML" COMPUTER NETWORKS, ELSEVIER SCIENCE PUBLISHERS B.V., AMSTERDAM, NL, vol. 33, no. 1-6, June 2000 (2000-06), pages 723-745, XP004304804 ISSN: 1389-1286 * abstract; figure 3 * * page 724, left-hand column, line 36 - right-hand column, line 19 * * page 724, right-hand column, line 36 - line 42 * * page 726, left-hand column, line 16 - page 727, left-hand column, line 23; figures 2,4,5 * * page 727, right-hand column, line 17 - line 24 * * page 730, right-hand column, line 8 - line 13 * * page 740, right-hand column, line 23 - line 39 * ---	1-37	G05F17/24
X	BRAGANHOLO V D P: "Updating relational databases through XML views" TECHNICAL REPORT RP-328, [Online] September 2002 (2002-09), XP002279067 UFRGS, Porto Alegre, RS, Brasil Retrieved from the Internet: <URL:http://www.inf.ufrgs.br/~vanessa/disciplinas/PropostaTese.pdf> [retrieved on 2004-05-04] * page 7, line 27 - line 41 * * page 13, line 6 - line 18 * * page 14, line 18 - line 20; table 2.1 * * page 15, line 23 - line 34 * * page 17, line 6 - page 18, line 8 * * page 19, line 12 - page 20, line 5; figures 3.1,3.2 * --- -/--	1-37	TECHNICAL FIELDS SEARCHED (Int.Cl.7) G05F

This present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
BERLIN	6 May 2004	Stauch, M

CATEGORY OF CITED DOCUMENTS

X: particularly relevant if taken alone
Y: particularly relevant if combined with another document of the same category
A: technological background
O: non-written disclosure
P: intermediate document

T: theory or principle underlying the invention
E: earlier patent document, but published on, or after the filing date
D: document cited in the application
L: document cited for other reasons

&: member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 04 00 2224

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
X	FALQUET G ET AL: "Design and Analysis of Active Hypertext Views on Databases" CUI - TECHNICAL REPORT, [Online] January 2002 (2002-01), XP002279068 Retrieved from the Internet: <URL:http://cui.unige.ch/isi/reports/design-anls-ahtv.pdf> [retrieved on 2004-05-04] * page 1, line 19 - page 2, line 7 * * page 3, line 19 - line 23 * * page 6, line 16 - line 28 * * page 19, line 17 - page 21, line 4 * ---	1-37	
A	CERI S ET AL: "DERIVING PRODUCTION RULES FOR INCREMENTAL VIEW MAINTENANCE" PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, 1994, pages 577-589, XP000914159 * page A * ---		
A	BONIFATI A: "Active Behaviors within XML Document Management" EDBT PH. D. WORKSHOP (EDBT PH.D. WS 2000), [Online] March 2000 (2000-03), XP002279069 Konstanz (Germany) Retrieved from the Internet: <URL:http://www.edbt2000.uni-konstanz.de/phd-workshop/papers/Bonifati.ps> [retrieved on 2004-05-04] * the whole document * ---		TECHNICAL FIELDS SEARCHED (Int.Cl.7)
A	US 6 460 860 B1 (MONDAY PAUL BRIAN) 12 November 2002 (2002-11-12) * abstract * -----		

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
BERLIN	6 May 2004	Stauch, M

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 04 00 2224

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

06-05-2004

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 6480860 B1	12-11-2002	NONE	

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82